Ebisuu_Training

Objetivo general

- Encender la máquina a través de **Integrations**.
    - Descripción:
        - Principalmente la plataforma ya no es la encargada de encender la máquina, esto queda a partir de ahora en manos de **Integrations**, dejando así en desuso las tareas celery que se tienen en ebisuu_training destinadas para esto.

Objetivos Especificos

- Crear un nuevo modelo para las propiedades de entrenamiento.
- Modificar vistas de Listado, Detalle y Creación para el módulo training.
- Crear api conector con **integrations**.

Actividades

- El módulo ebisuu training fué modificado agregando un nuevo campo foráneo (training_property) al modelo **ModelTrainig**

- Se creó un nuevo modelo **TrainingProperty**, este nuevo modelo va tener los siguientes campos:

    - transaction_id
    - name
    - epochs
    - batch
    - split_train
    - backbone
    - augmentation_policy
    - use_data_augmentation
    - user
    - is_active
    - pub_date
    - mod_date

- Se creó una nueva carpeta **utils** dentro del modulo ***ebisuu_training***, esta tiene dos scripts python:

    - ***token_integrations.py***: dentro está la función `get_token_integrations`, esta sirve para Autenticarse en integrations, esta recibe unicamente el username que esté autenticado en la plataforma.
    - ***integration_training.py***: dentro está la función `init_integration_training`, esta sirve para Enviar parámetros a la api de integracion que de inicio a un nuevo entrenamiento, recibe un primer parámetro que es el username que al igual que en la función anterior es del `request.user`, siendo `request.user.username` para ambos caso y por ultimo un diccionario `**kwargs`.

- La vista para crear nuevos entrenamientos sufrió un par de modificaciones:

    - Se agregó un nuevo campo al modelo, esto implica que se garantizó la integridad referencial, pues al momento de guardar un nuevo objeto esta asociación se realiza sin problema alguno.
    - Ya no se usa celery, si no que sencillamente se va a enviar los parametros a **Integrations** a través de api, en donde mandamos los parámetros acostumbrados.

```
    # aqui debemos comunicarnos con integrations con celery
    username: str = request.user.username
    init_integration_training(
        username, **array_data
        )
```

- La vista que lista los modelos de entrenamientos se les fué agregado una nueva columna que muestra las propiedades asociadas.
- La vista de detalle ya no muestra los campos que anteriormente mostraba con respecto a las propiedades, esta se extraen de unas `@property` que se crearon en el modelo para tocar lo menos posible estos templates.

```
    @property
    def epochs_(self):
        return f"{self.training_property.epochs}"

# Ejemplo de property del modelo
```